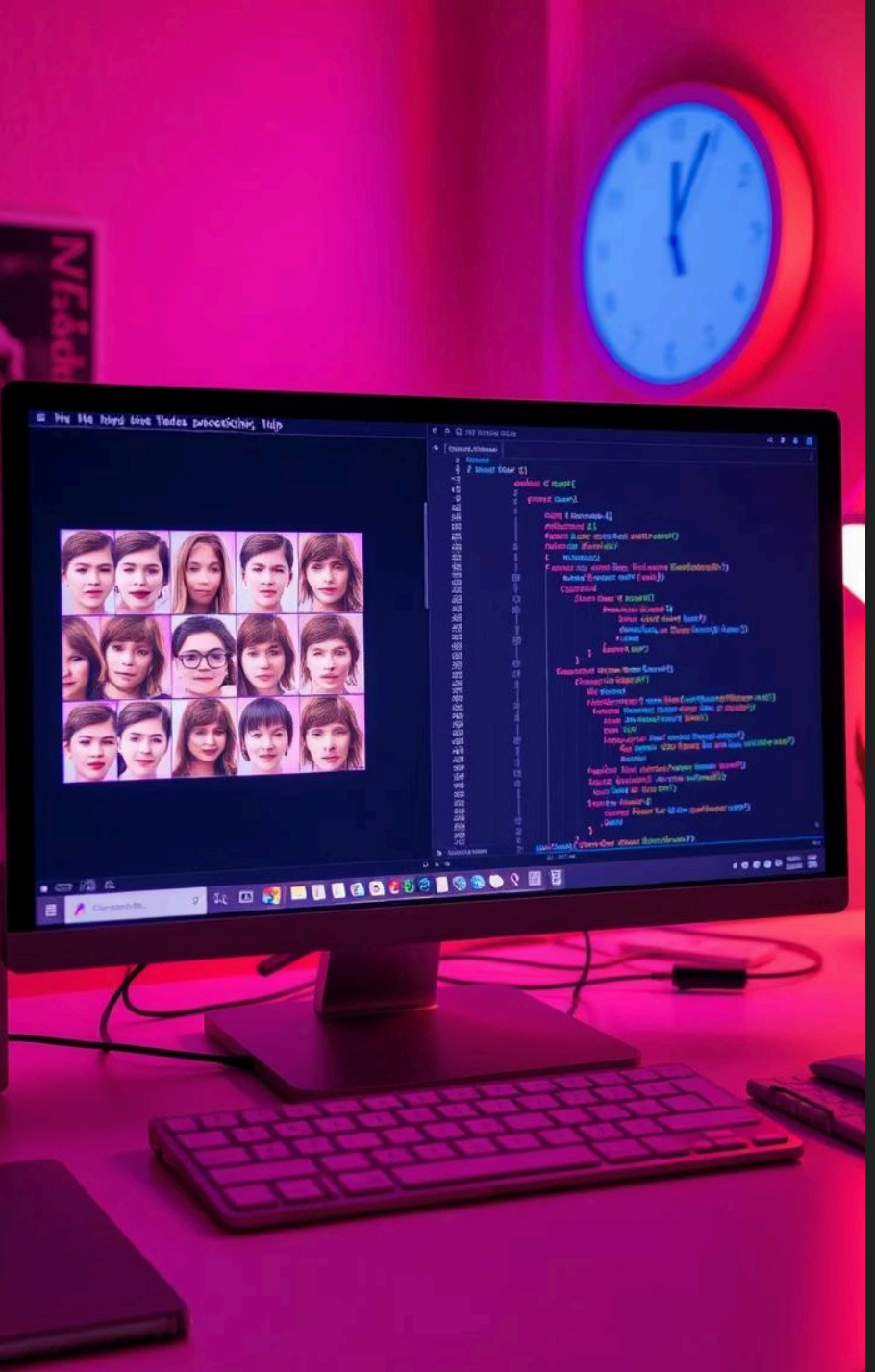
Emotion Detection from Facial Images

Field: Human Emotion Detection from Faces

Motivation: Emotion detection plays a crucial role in various applications, including:

- **Human-Computer Interaction (HCI):** Improving user experience in AI assistants, gaming, and virtual reality.
- **Mental Health Analysis:** Assisting psychologists in tracking emotional states in patients.
- **Surveillance & Security:** Detecting suspicious behavior in public spaces.
- **Entertainment & Marketing:** Analyzing audience reactions to ads or movies.

This project aims to classify facial emotions in static images using **Computer Vision (CV)** and **Deep Learning (DL)** techniques, comparing two approaches: **Convolutional Neural Networks (CNN)** and **Support Vector Machines (SVM)**.



Dataset and Why It Was Selected

Dataset: FER2013 (Facial Emotion Recognition 2013)

- **Source:** Kaggle
- **Classes:** 7 emotions (**Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral**)
- **Size:** 35,000 grayscale images (48×48 pixels)
- **Split:** Training (80%), Validation (10%), Testing (10%)

Why FER2013?

✔️ **Standard Benchmark:** Widely used in emotion recognition research.

✔️ **Diverse Expressions:** Covers common emotions with real-world variations.

✔️ **Preprocessed Format:** Images are already aligned and cropped, reducing preprocessing complexity.

Pipeline: Step-by-Step Explanation

• Data Preprocessing & Image Enhancement

Input: Raw facial images (48×48 grayscale).

Output: Clean, normalized images ready for segmentation.

Steps:

1. Grayscale Conversion (it is already grayscale).

2. Noise Reduction (Non-local Means Denoising).

3. Contrast Enhancement (Contrast Limited Adjustment Histogram Equalization).

4. Blurring (Apply Gaussian Blurring on the image).

5. Sharpening (Sharpens the image by subtracting a blurred version (Unsharp Masking)).

6. Normalization (Pixel values scaled to [0, 1]).

Justification:

• Using MTCNN ensures robust and accurate face detection, especially in images with multiple faces or varying orientations. It extracts only the relevant region of interest (the face) instead of the full image, reducing noise from backgrounds.

• Cropping focuses the model on facial features only, which are the key indicators for emotion classification. It eliminates distractions like background objects or clothing.

• Standardizing all face images to 48×48 ensures uniform input dimensions for neural networks and aligns with the original FER2013 dataset's format. This consistency is crucial for model convergence and efficiency.

• Removing color information reduces input complexity, speeds up training, and prevents color from misleading the model, since emotions are primarily expressed via **shape** and **texture**, not color.

Follow Pipeline:

•Segmentation (Face Detection & Cropping)

Input: Preprocessed grayscale images.

Output: Cropped facial regions for each detected face.

Steps:

1. Face Detection using MTCNN (MTCNN identifies the bounding box around the face and provides key facial landmarks).

2. Face Cropping (For each detected face, only the facial region (inside the bounding box) is extracted from the original image. This is the actual segmentation step, where we separate the face from the background).

3. Image Resizing (The cropped face is resized to 48×48 pixels, which matches the expected input size of the FER2013 dataset and most CNN models).

4. Grayscale Conversion (The resized face image is converted to grayscale to reduce complexity and focus on facial structure rather than color).

5. Apply preprocessing

Justification:

• Preprocessing standardizes the data format and improves model convergence and stability during training

• CLAHE improves local contrast without amplifying noise.

• Sharpening helps in better edge detection for feature extraction

• Augmentation balances between categories so the model learns fairly and Improved model accuracy. and reduce overfitting.

Follow Pipeline:

• Feature Extraction

Two Approaches:

A. For SVM (Handcrafted Features)

Input: Cropped, grayscale facial images (48×48).

Output: Feature vector (HOG).

Techniques:

HOG (Histogram of Oriented Gradients): Extract **HOG features** to capture structural information in the face. This helps in identifying key facial structures such as eyes, nose, and mouth.

HOG features focus on gradient-based edge information, which is crucial for distinguishing facial features.

Justification:

• **HOG** is robust to illumination changes.

B. For CNN (Convolutional Neural Networks)

Input: Cropped, grayscale facial images (48×48).
Output: Deep features extracted via convolutional layers.

Justification:

• CNNs automatically learn hierarchical features (edges → textures → facial components).

• Better accuracy than handcrafted features for complex patterns.

Follow Pipeline:

· Data Augmentation (Handling Class Imbalance)

Problem: FER2013 is imbalanced (e.g., more "Happy" than "Disgust" samples).
Solution: Generate synthetic data using:

· Rotation (±10°), Shifting (10%), Zooming (10%), Flipping (Horizontal).

Why?
✔️ Prevents model bias toward majority classes.
✔️ Improves generalization.
✔️Augmentation balances between categories so the model learns fairly and Improved model accuracy.
and reduce overfitting.

Before vs. After Augmentation:

Class	Original Count	After Augmentation
Sad	2922	3,500
Disgust	262	3,500

Follow Pipeline:

• Model Classification

Input:

Preprocessed face images cropped using MTCNN, Images are resized to **48×48 grayscale** format.

For SVM: input is the **HOG feature vector** extracted from these images (900 features per image).

For CNN: input is the **48×48 grayscale images** (as raw pixel data).

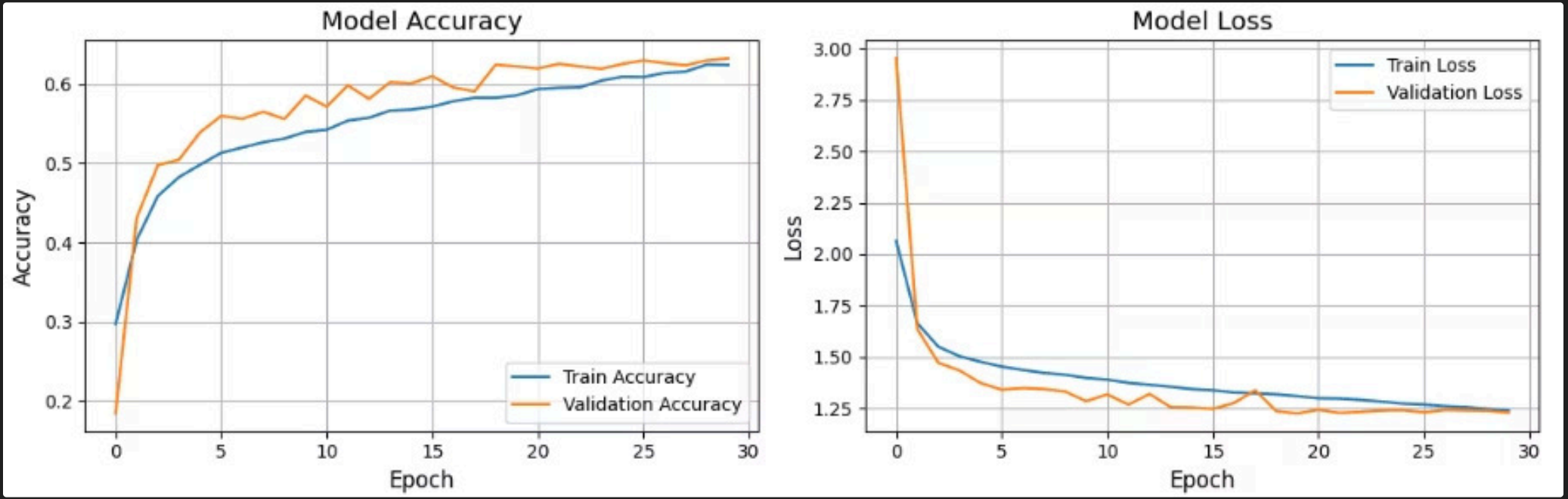
Output: Predicted emotion labels for each input image

Steps:

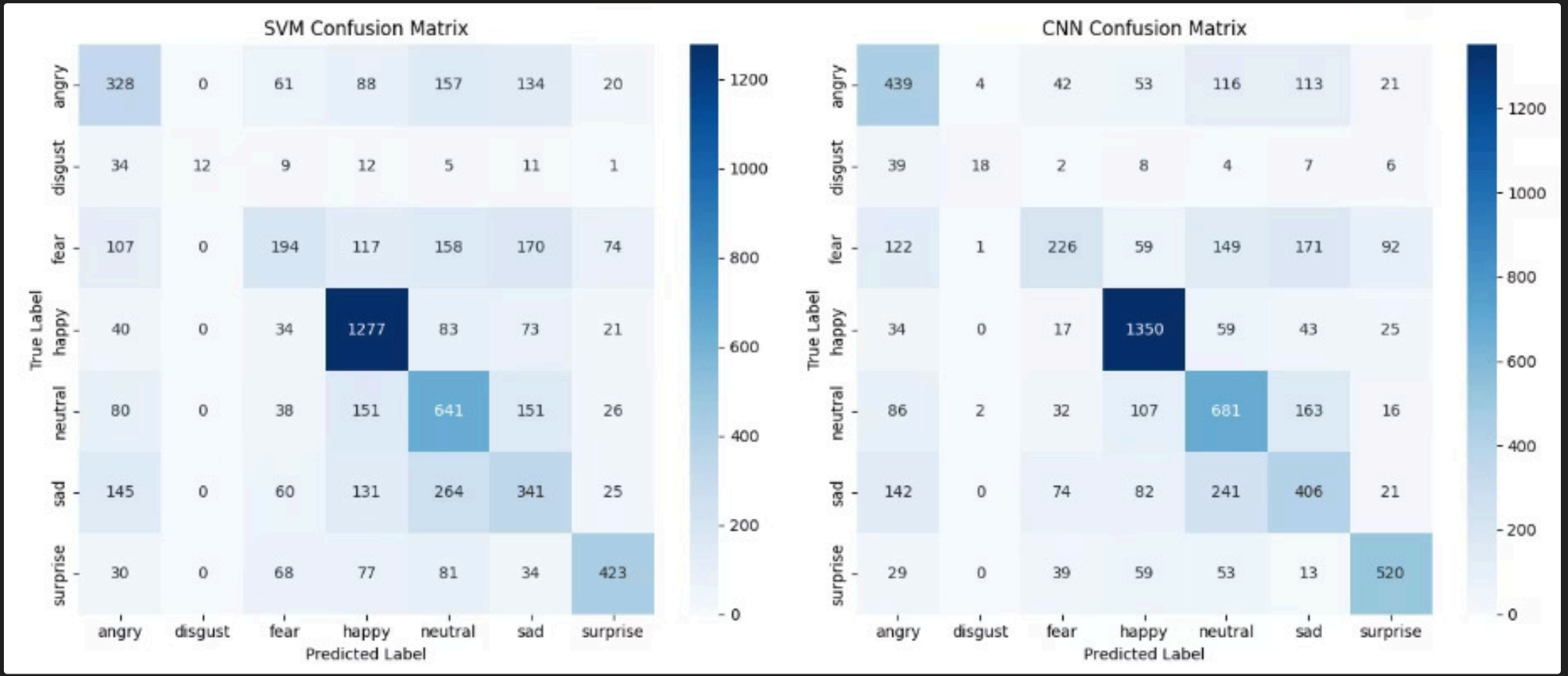
1. Dataset Preparation Split the preprocessed dataset into Training set & Validation set & Test set.
2. Performed data augmentation to balance the classes.
3. Feature Extraction (for SVM only) Used HOG (Histogram of Oriented Gradients) to convert each image into a 900-dimensional feature vector.
4. Model 1: SVM Trained a Support Vector Machine with RBF kernel using HOG features and Evaluated performance using: Accuracy & Precision, Recall, F1-score & Confusion Matrix.
5. Model 2: CNN Built a CNN with: Multiple convolutional + pooling layers & BatchNormalization + Dropout to prevent overfitting & Fully connected layer and a softmax output layer & Used callbacks: EarlyStopping, ReduceLROnPlateau, ModelCheckpoint & Trained the model on the augmented training set & Evaluated on validation and test sets & Evaluation Computed: Training and validation accuracy/loss & Test accuracy & Precision, Recall, F1_score & Confusion matrix visualization.

Models Evaluation and visualization:

Model accuracy and loss (CNN)

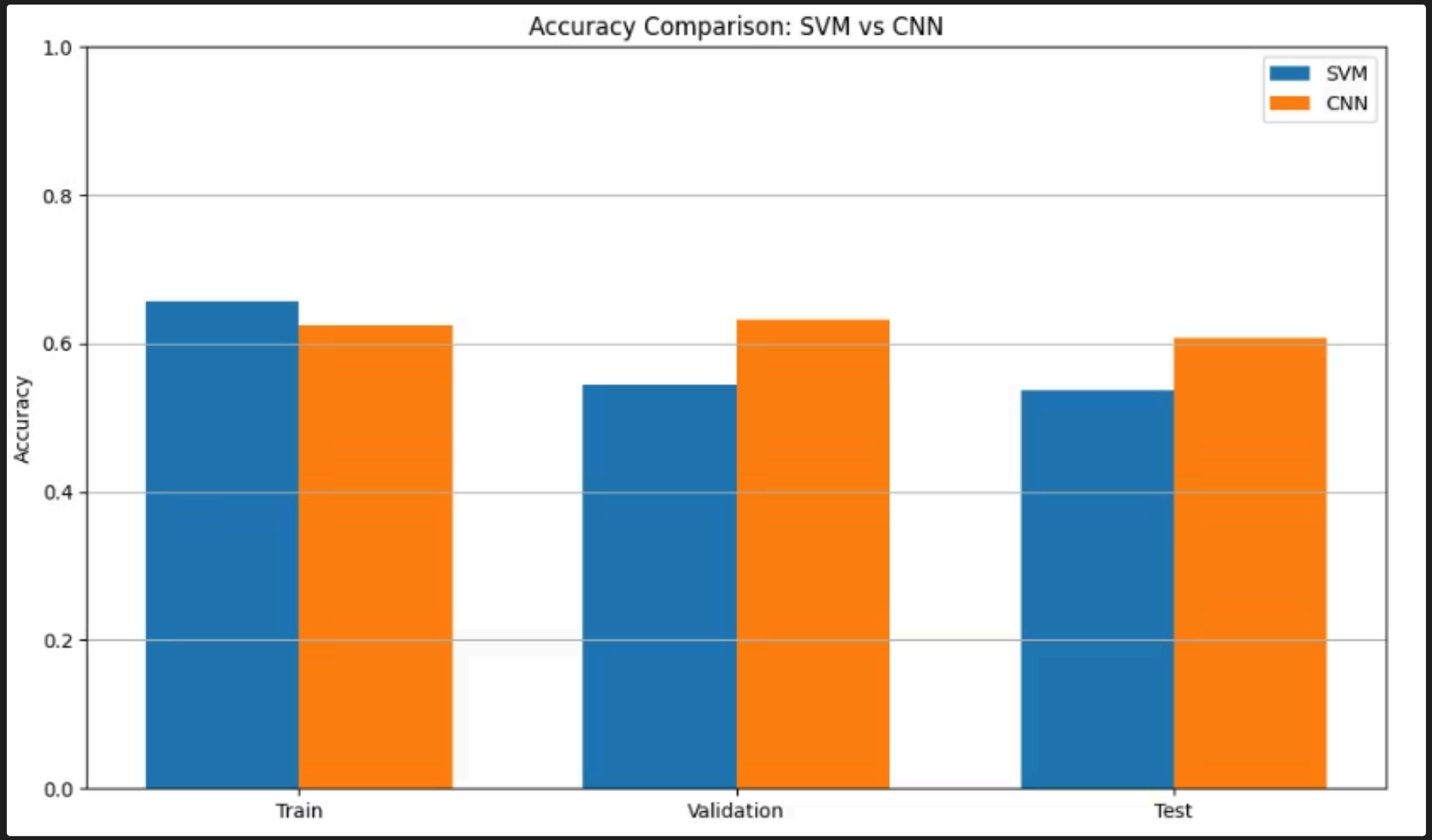


confusion matrix (SVM) VS (CNN)

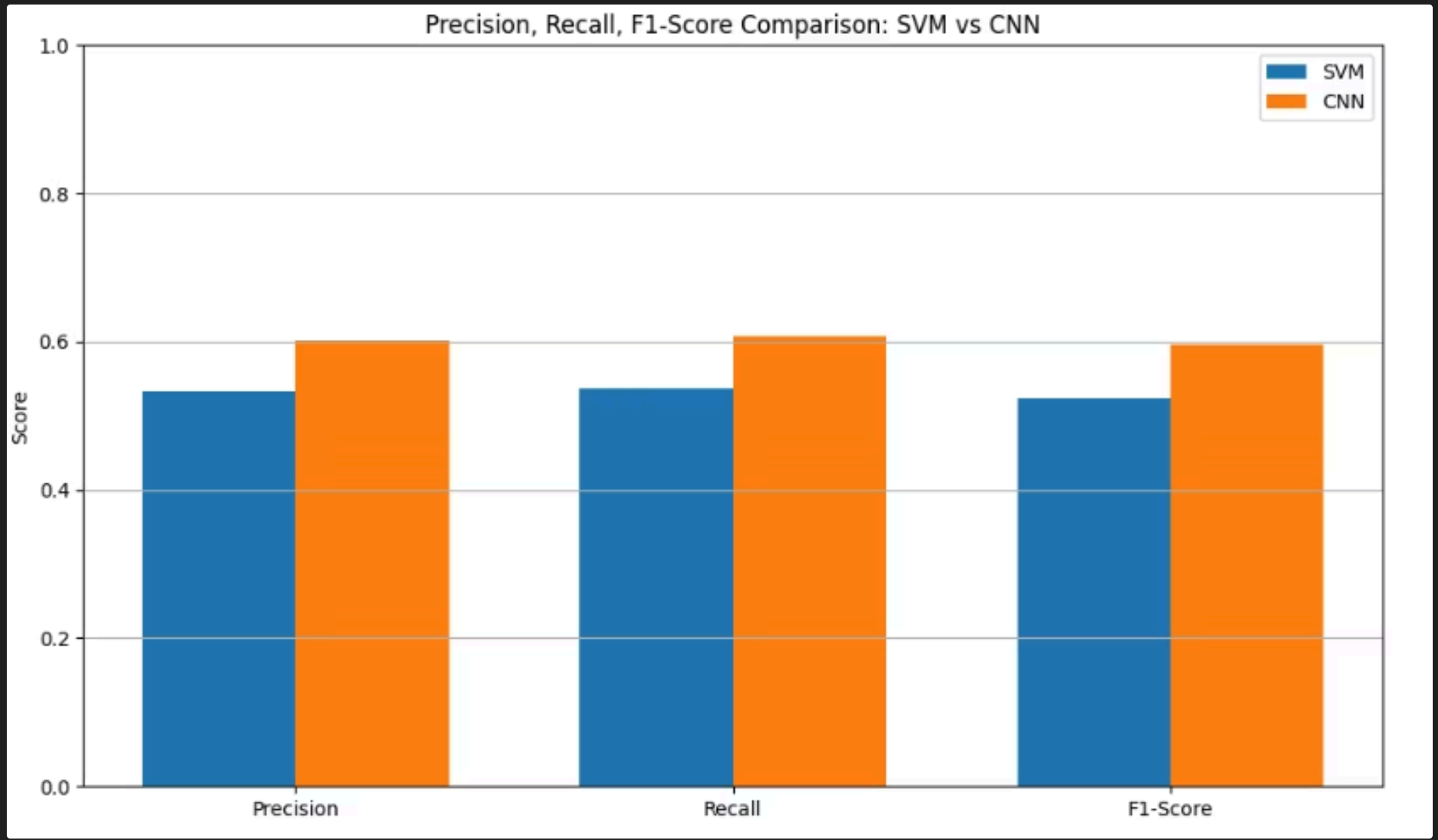


Metric	SVM	CNN
Train Acc	0.6573	0.6244
Val Acc	0.5446	0.6326
Test Acc	0.5373	0.6081
Precision	0.5331	0.6020
Recall	0.5373	0.6081
F1-Score	0.5233	0.5960

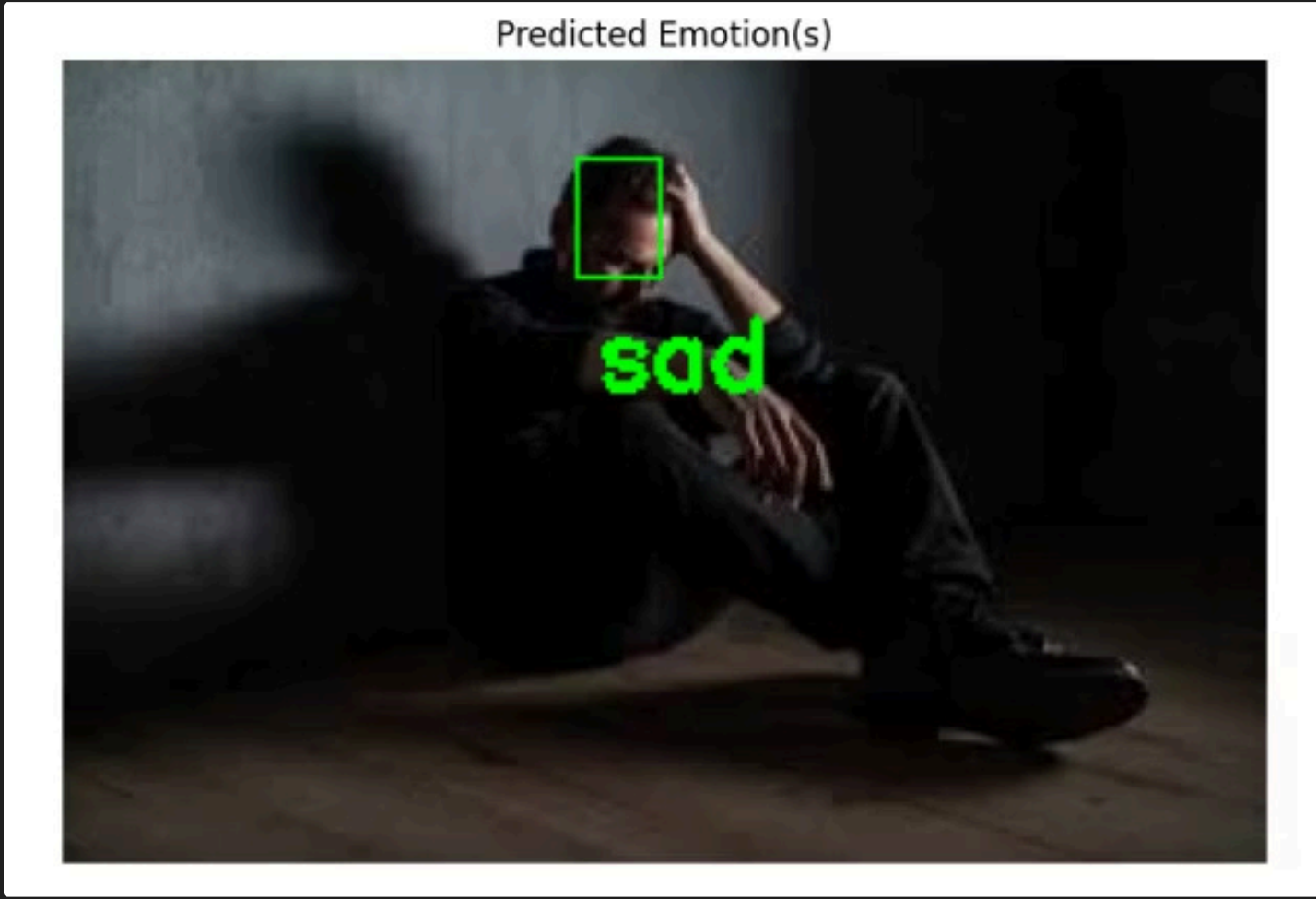
Accuracy comparison (SVM) VS (CNN)



Precision, Recall, F1_Score comparison (SVM) VS (CNN)



Visual example for prediction of an external image



Conclusion

Limitations and Challenges

· FER-2013 is grayscale and low-res

· Performance drops with occlusion, lighting, or angles

· After experimenting with data augmentation, we found that having around 3500 images per class gave us the best results. Increasing or decreasing this number led to a noticeable drop in accuracy.

References

Goodfellow, I., et al. (2013). *Challenges in Representation Learning: FER2013*.

Thank You

Questions and discussion welcome.